

Fall 12-18-2018

# DEEP VISUAL RECOMMENDATION SYSTEM

Raksha Sunil  
*San Jose State University*

Follow this and additional works at: [https://scholarworks.sjsu.edu/etd\\_projects](https://scholarworks.sjsu.edu/etd_projects)

Part of the [Artificial Intelligence and Robotics Commons](#)

---

## Recommended Citation

Sunil, Raksha, "DEEP VISUAL RECOMMENDATION SYSTEM" (2018). *Master's Projects*. 659.

DOI: <https://doi.org/10.31979/etd.gf9c-u3s8>

[https://scholarworks.sjsu.edu/etd\\_projects/659](https://scholarworks.sjsu.edu/etd_projects/659)

This Master's Project is brought to you for free and open access by the Master's Theses and Graduate Research at SJSU ScholarWorks. It has been accepted for inclusion in Master's Projects by an authorized administrator of SJSU ScholarWorks. For more information, please contact [scholarworks@sjsu.edu](mailto:scholarworks@sjsu.edu).

# DEEP VISUAL RECOMMENDATION SYSTEM

A Project Report

Presented to

Dr. Robert Chun  
Department of Computer Science  
San Jose State University

In Partial Fulfillment  
Of the Requirements for the Class  
CS 298

By  
Raksha Sunil  
December 2018

## **Abstract**

Recommendation system is a filtering system that predicts ratings or preferences that a user might have. Recommendation system is an evolved form of our trivial information retrieval systems. In this paper, we present a technique to solve new item cold start problem. New item cold start problem occurs when a new item is added to a shopping website like Amazon.com. There is no metadata for this item, no ratings and no reviews because it's a new item in the system. Absence of data results in no recommendation or bad recommendations. Our approach to solve new item cold start problem requires only an image of a new item. A deep learning architecture is used to extract feature vector from an image. Using a distance metric, the distance between various image feature vectors are calculated. Finally, the model recommends most similar items to the users.

## **Acknowledgements**

I would like to take this opportunity to thank my project advisor, Dr. Robert Chun, for his constant support and guidance through this journey. I would also like to extend my thanks to my committee members, Dr. Thomas Austin and Shравanthi Pachalla for their time, participation and feedback. And last but not the least my family, friends and everyone who were part of this special journey for their constant support and encouragement. Without whom this wasn't possible.



## Table of Contents

<b>CHAPTER 1</b> .....	<b>5</b>
1.1 INTRODUCTION .....	5
1.2 BACKGROUND.....	6
1.3 SUPERVISED LEARNING.....	7
1.4 UNSUPERVISED LEARNING .....	9
1.5 REINFORCEMENT LEARNING, HYBRID MODELS AND BEYOND.....	11
<b>CHAPTER 2</b> .....	<b>13</b>
2.1 RECOMMENDATION TECHNIQUES .....	13
2.1.1 <i>Explicit Data or Explicit Ratings</i> .....	15
2.1.2 <i>Implicit Data or Implicit Ratings</i> .....	15
<b>CHAPTER 3</b> .....	<b>17</b>
3.1 CHALLENGES IN RECOMMENDATION SYSTEMS .....	17
3.1.1 <i>Cold Start and Sparsity</i> .....	17
3.1.2 <i>Over Specialization</i> .....	17
3.1.3 <i>Privacy</i> .....	18
3.1.4 <i>Scalability</i> .....	18
3.1.5 <i>Freshness or a Change in Prediction</i> .....	18
<b>CHAPTER 4</b> .....	<b>19</b>
4.1 HANDLING COLD START PROBLEMS.....	19
4.2 NEW USER COLD START PROBLEM .....	19
4.2.1 <i>Demographic Information</i> .....	19
4.2.2 <i>Requesting Users to Provide Ratings</i> .....	21
4.2.3 <i>Asking Users to Select Few Users That They Trust</i> .....	25
4.2.4 <i>Tags</i> .....	26
4.3 COLD START ITEM PROBLEM .....	26
<b>CHAPTER 5</b> .....	<b>28</b>
5.1 DEEP LEARNING .....	28
5.2 DEEP LEARNING ARCHITECTURES .....	29
5.2.1 <i>Applications of Deep Learning Architectures</i> .....	29
5.3 VISUALIZING FEATURES OF IMAGES.....	30
<b>CHAPTER 6</b> .....	<b>33</b>
6.1 PROPOSED TECHNIQUE .....	33
6.2 DATA COLLECTION.....	34
6.3 DATA PROCESSING .....	34
6.3.1 <i>Resizing an Image</i> .....	35
6.4 MODEL.....	36
6.4.1 <i>Architecture of VGG19</i> .....	38
6.4.2 <i>Transfer Learning</i> .....	38

<b>CHAPTER 7</b> .....	<b>40</b>
7.1 BASELINE FOR EVALUATION .....	40
7.2 EVALUATION METRIC .....	40
7.3 HARDWARE AND SOFTWARE REQUIREMENTS .....	41
7.4 DATASET 1 .....	42
7.5 DATASET 2 .....	45
7.6 DATASET 3 .....	49
7.7 DATASET 4 .....	52
CONCLUSION .....	56
<b>REFERENCES</b> .....	<b>58</b>

## List of Tables

Table 1 Example of image parameters in an image .....	35
Table 2 Baseline for our research .....	40
Table 3 Experimental results of Dataset 1 .....	43
Table 4 Experimental results of Dataset 2 .....	47
Table 5 Experimental results of Dataset 3 .....	50
Table 6 Experimental results of Dataset 4 .....	54
Table 7 Comparison of our experiments with the baseline approach .....	56

## Table of Figures

Fig. 1. Conceptual map of research .....	6
Fig. 2. Classification and Regression examples.....	8
Fig. 3. Clustering example.....	10
Fig. 4. Example model of reinforcement learning.....	11
Fig. 5. Collaborative filtering.....	13
Fig. 6. Content-based filtering .....	14
Fig. 7. Rating of an item .....	15
Fig. 8. Review of an item.....	16
Fig. 9. Demographic information example.....	20
Fig. 10. Results of A. M. Rashid et al. technique [9] .....	24
Fig. 11. Example of basic neural networks.....	28
Fig. 12. An example of multi object detection in computer vision.....	29
Fig. 13. Example of single neuron feature visualization. From left to right edges, textures and patterns from [16] .....	30
Fig. 14. Example of single neuron feature visualization. From left to right object parts and objects from [16].....	30
Fig. 15. Visualization of interaction between two neurons from [16] .....	31
Fig. 16. Example of feature visualization from [16] .....	32
Fig. 17. Example of feature attribution from [16] .....	32
Fig. 18. An example of an image after resizing process .....	36
Fig. 19. Evaluation Formulae .....	41
Fig. 20. Glimpse of Dataset 1 .....	42
Fig. 21. From top to bottom, the best and worst circle image recommendations given by our model with k=5, algorithm=brute and metric=cosine .....	45
Fig. 22. Glimpse of Dataset 2.....	46
Fig. 23. From top to bottom, the best and worst onion rings image recommendations given by our model with k=5, algorithm=brute and metric=cosine.....	48
Fig. 24. Glimpse of Dataset 3.....	49
Fig. 25. From top to bottom, the best and worst hotdog image recommendations given by our model with k=5, algorithm=auto and metric=cosine .....	52
Fig. 26. Glimpse of Dataset 4 .....	53
Fig. 27. From top to bottom, the best and worst taiga image recommendations given by our model with k=5, algorithm=auto and metric=cosine .....	55

## Chapter 1

### 1.1 Introduction

Web users these days have access to data available in huge databases. This leads to the popularity of systems known as recommendation systems that recommends an item to solve the problem of plenty [1]. A recommendation system in general is a technique that recommends a likable item/items to the user. Hence a recommendation system plays a vital role for both online business providers and their users. Techniques such as Collaborative Filtering (CF) assume that there is enough user preference information readily available in order to create personalized recommendations [2]. A recommendation system requires the users to create a profile and fill in the profile information, or it will require users' usage history. Another technique is to explore available information related to metadata of items to predict recommendations [3]. These techniques assume that user information, metadata of an item and item features are always available, which is not true.

In this paper, a recommendation system is proposed which takes a selected item as an input and recommends an item/items as the output. The goal is to precisely predict and recommend items to the user even when his/her profile information is unavailable. The absence of user information is a serious real world problem, and it belongs to a special category of problem known as the cold start problem [4].

In this research, we will explore various types and issues of recommendation systems and propose a technique to handle new item cold start problems. The research is organized as shown in Fig. 1. We aimed to address the following questions:

RQ1: What are the various types of recommendation systems?

RQ2: Challenges in recommendation systems?

RQ3: What are the current recommendation approaches to solve them?

RQ4: How to solve cold start problem?

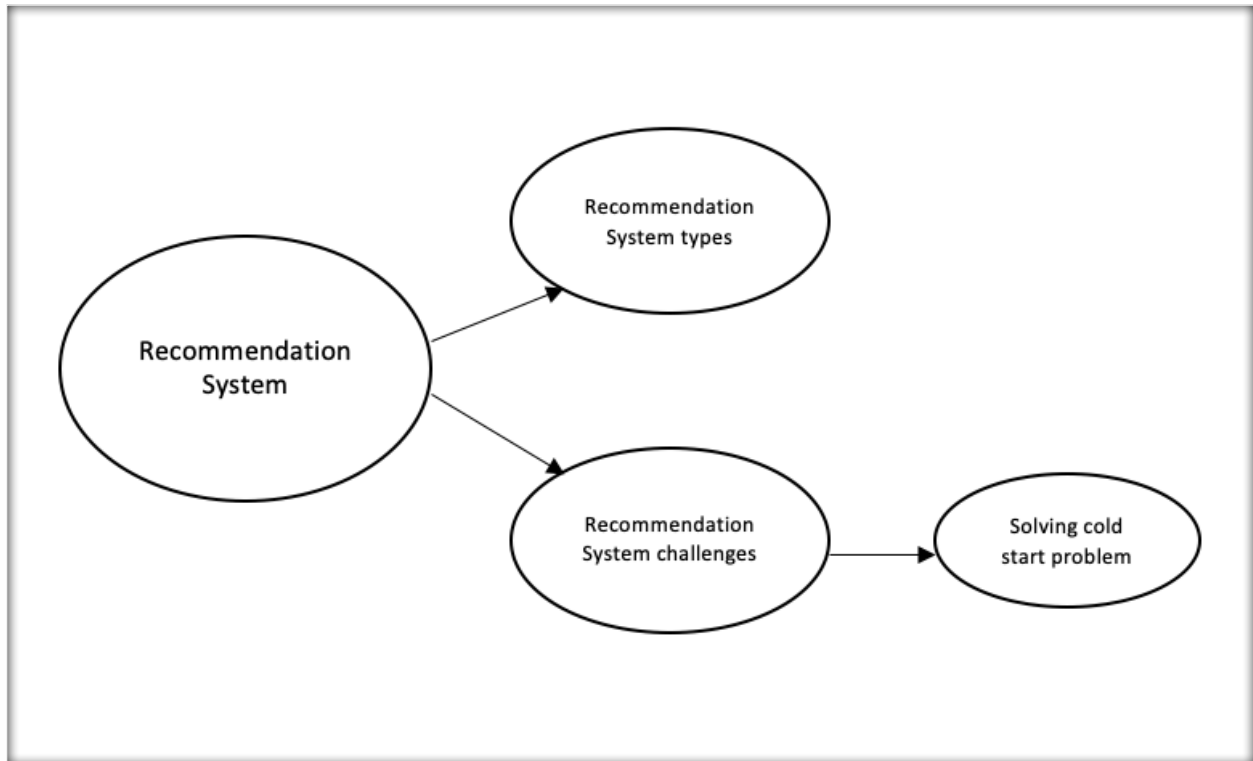


Fig. 1. Conceptual map of research

## 1.2 Background

Machine Learning has been in the main stream of information technology since the last two decades and it has been involved in our lives one way or the other way. As the population

using technology exploded there was a lot of data created by people. Analyzing this massive data became the main ingredient of technology evolution. Hence machine learning became popular and new machine learning algorithms are born almost every day.

Machine learning is broadly classified into two categories: supervised and unsupervised learning. In supervised learning, the computer has a general rule for connecting inputs to outputs. The output is usually referred to as the target variable or target label. In unsupervised learning, the target variable is not explicitly given. The machine learning algorithm has to figure out the hidden pattern between the inputs and outputs. This can be considered as one of the challenges in unsupervised learning. Recommendation systems are supervised learning, unsupervised learning and sometimes both. The category of the recommendation system entirely depends on the area and scope of the application. When machine learning algorithms were not able to solve a certain real world problem, neural networks also known as deep learning were used. We will discuss briefly about each one of them in the next sections.

### **1.3 Supervised Learning**

Supervised learning is a class of algorithms where most of our real world problems that are straightforward simple tasks fall under this category. In this type of learning, the algorithm will have access to data with correct input - output pairs which can be used to train the algorithm during its training phase. A common example is handwriting recognition. We typically approach this problem as a supervised learning task. In this task, we train the algorithm with various number of handwritten images with correct labels and the algorithm learns the relationship between images and their respective correct labels as a pattern and will remember it. Then when

we send a new image to the algorithm, it will recognize the handwritten pattern and predict its label.

Supervised learning is an easy task because the algorithm will learn using explicit examples and it is also easy for humans to understand. But supervised learning can be done only when there is data available. The data should be labeled too, which means at some point of time a human would have collected data, processed them, arranged them in a structured format and labeled them correctly. This is a tedious process but once we have labeled data then it's very easy to deal with supervised learning tasks. Many real time problems fall into this category of problems.

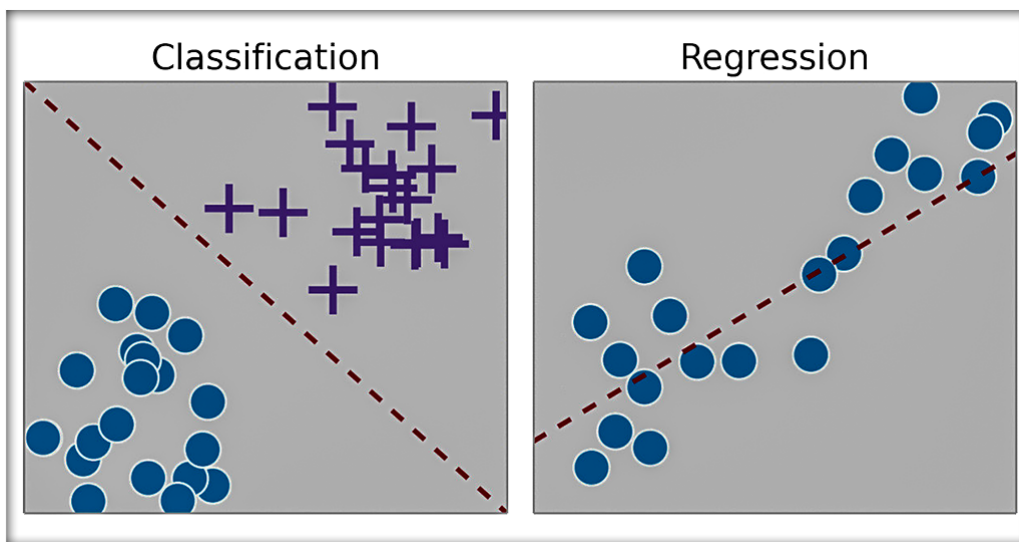


Fig. 2. Classification and Regression examples

Supervised machine learning algorithms can be further classified into regression and classification tasks as shown in Fig. 2. A regression problem is a problem where the output variable is a real continuous value, such as stocks, weight and currency value. To solve such



problems using supervised learning method, we will need historical data related to the problem. The algorithm will learn the pattern between input and output variables to predict the continuous outcomes for the current input variable. A classification problem is a problem where the output variable is a discrete value, mostly a category or belongs to a class. For example, size of a fruit - big, medium or small, color of a vegetable - red, yellow or purple, does the given patient have breast cancer - yes or no. If a problem has only two classes, then it's called binary classification. If a problem has more than two classes, then the problem is called multiclass classification. Since classification is a supervised learning algorithm, it learns the pattern using historical examples and classifies a new input variable into one of the classes.

## **1.4 Unsupervised Learning**

On the other end is unsupervised learning. It is an entirely different class of algorithm. Supervised learning algorithms find and learn patterns using a dataset containing right input – output pairs but in unsupervised learning, the algorithms find and learn patterns from unlabeled data. This may happen because of the unobservable nature of the problem; its infeasible or sometimes there is no right answer for a given problem.

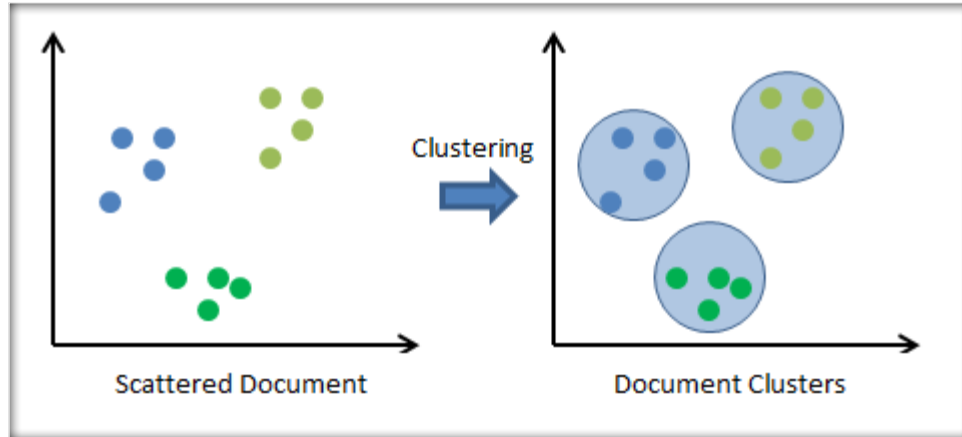


Fig. 3. Clustering example

Unsupervised learning can be classified into clustering and association rules. Clustering is a task where similar observations are grouped together as shown in Fig. 3. Observations in different groups have different behavior. For example, customers are put into different clusters based on their similar buying habits. Customers within the cluster have similar buying habits and customers from different clusters have dissimilar buying habits. Unsupervised learning algorithm plays a prominent role in solving sales related problems. The difficulty in such problems is that we never know how many clusters the customers should be divided into. In other words, we do not know how the clusters should look like. The most popular algorithm used for forming clusters is K-means algorithm; where K is equal to the number of clusters to be formed.

Association rule problem is a problem where one has to discover the association between the inputs and learn the hidden pattern. This usually describes a large portion of the data, such as customers who bought this item may have also bought this. For example, customers who buy milk also tend to buy eggs and bread. For this reason, you will find milk, eggs and bread placed close to each other in all the supermarkets.

## 1.5 Reinforcement Learning, Hybrid Models and Beyond

This type of problem has gained importance recently due to its nature of learning. In this type of problem, we will not provide the algorithm with historical data examples with correct input - output pairs, but what we will provide is a procedure as shown in Fig. 4. for the algorithm to quantify its performance in the form of a reward signal. Reinforcement learning resembles how humans, animals and other living things learn. When the algorithm tries a bunch of new and different things, it is rewarded when it does something well.

Reinforcement learning can be applied to problems where it's solution space is infinite. One real world example of reinforcement learning is a small group of people who came up with an algorithm to play Atari games. The model was well trained and it outperformed all the human players and later the creators sold the company to Google for \$500 Million. If we choose to implement this solution using a supervised learning, it will require an extremely large dataset

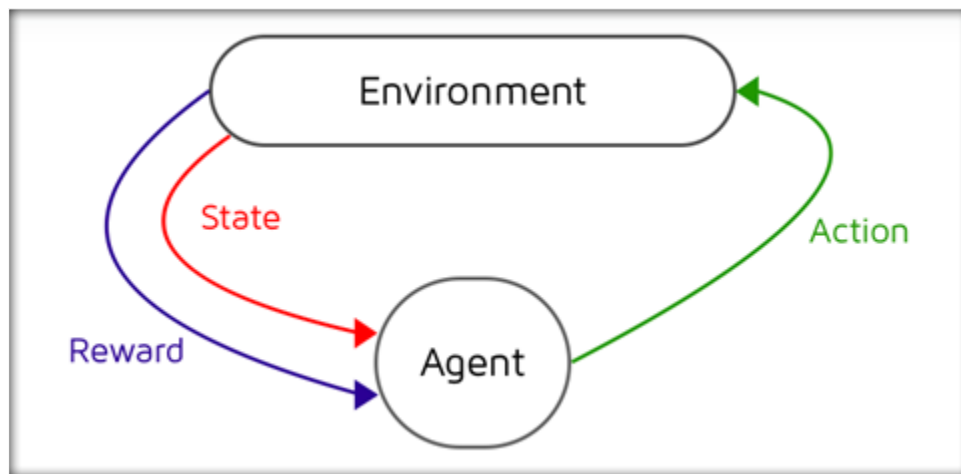


Fig. 4. Example model of reinforcement learning

containing human and game interactions. But if we use the reinforcement learning technique to solve this problem it will not require any data at all. The algorithm itself creates its own data. And the algorithm rewards itself when it performs well. Reward is usually in the form of a score. The higher the score the better the model performance. The ultimate goal of this algorithm will be to how to maximize the score.

Hybrid algorithms are often a combination of supervised and unsupervised learning algorithms. For example, let's consider the anomaly detection problem which is a common problem in various fields. Here an anomaly is detected when something unexpected occurs in a usual pattern. The pattern can be remembered by the algorithm using supervised learning and when it sees something unusual it will recognize it as abnormal behavior and categorize it into different sections. This can be achieved by using unsupervised learning.

## Chapter 2

### 2.1 Recommendation Techniques

Collaborative filtering, content based filtering and hybrid filtering are the 3 main types of filtering techniques used in recommendation systems. Users' past interactions and similar users' behavior are recorded in collaborative filtering as showing in Fig. 5. This information will be used to recommend an item to the user.

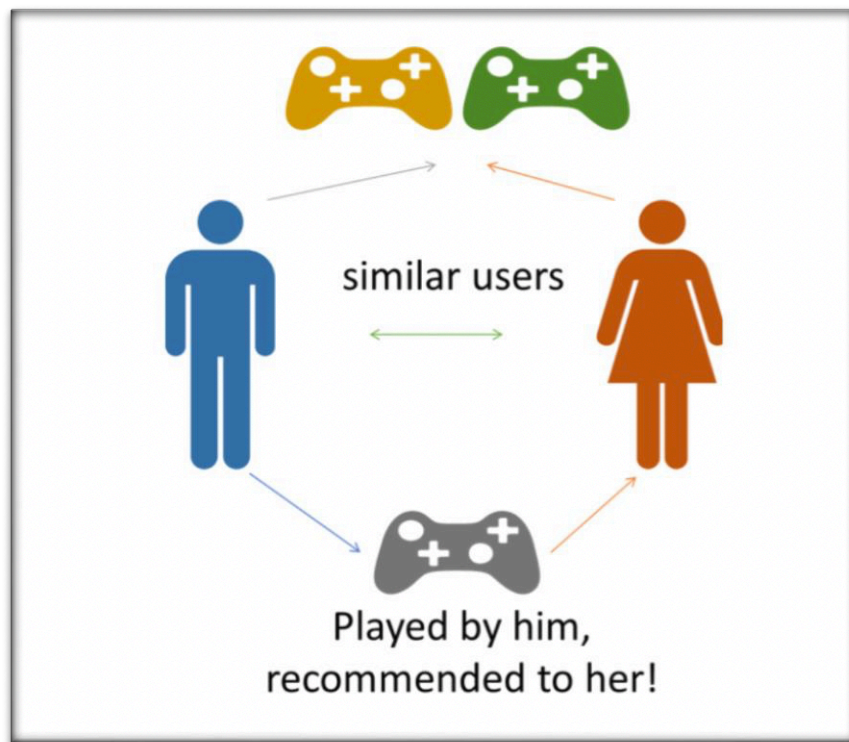


Fig. 5. Collaborative filtering

Users' item characteristics are used to determine which item to recommend to the user in content based filtering as shown in Fig. 6. And in hybrid recommendation systems,

collaborative filtering and content based filtering are combined. These techniques assume that the user information is always available, but in the real world the user information is not always available. Overcoming the unavailability of user information is the biggest challenge in recommendation systems. There is a lot of active research going on to find ways to handle cold start problems and we will discuss a few of these studies in the upcoming chapters.

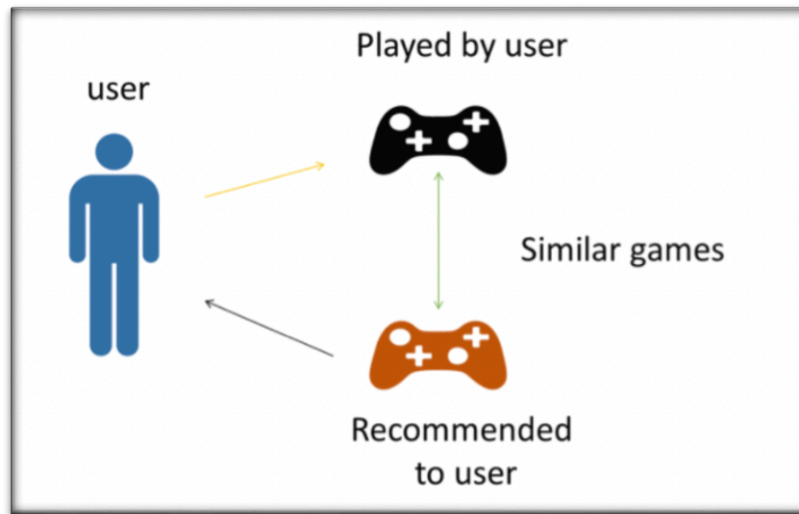


Fig. 6. Content-based filtering

Research has proven that a hybrid recommendation technique outperforms content-based and collaborative filtering in special cases. Hybrid systems can be implemented in various ways such as - by combining these techniques into one technique, by using the predictions of content based and collaborative filtering techniques and later combining them for a final result or by simply adding collaborative based capabilities to a content based technique and vice versa. Hybrid techniques can be used to solve a few challenges of recommendation systems like the sparsity problem. Best examples of hybrid recommendation systems in the real world are

YouTube recommendation and Netflix recommendation systems.

Before discussing more about recommendation systems in detail we'll discuss the types of ratings given to an item by the users. These ratings are the main component of recommendation systems. Using these ratings given by the users, the recommendation system recommends items to the user. Ratings collected by recommendation systems in a shopping website such as ebay.com, amazon.com are broadly classified as follows.

### 2.1.1 Explicit Data or Explicit Ratings

Explicit data in a recommendation system is collected by asking the users to rate items on a scale of 1 to 5. This type of rating is generally measured in stars, 1 star being low rating and 5 stars being highest rating. Thus, if user rates an item with more stars then they are more likely to like it. This data is stored and used by the recommendation system to recommend items to the users. For example, my favorite cookie is rated with 5 stars on Amazon.com as seen in Fig. 7.



Fig. 7. Rating of an item

### 2.1.2 Implicit Data or Implicit Ratings

Implicit data in a recommendation system is collected by asking the users to review an

item they purchased. If a user had good experience with the item, the review of that item is going to be positive, else the review is negative. We can determine the emotion of the reviews by using sentimental analysis. Reviews given by the users to the items in shopping websites are stored by the online business providers and later used for recommending items to the users in the future.

For example, review of my favorite cookie on amazon.com as shown in Fig. 8.



Fig. 8. Review of an item



## Chapter 3

### 3.1 Challenges in Recommendation Systems

In paper [5], the author talks about important and popular challenges in online shopping recommendation systems and lists an example for each.

#### 3.1.1 Cold Start and Sparsity

We will be combining cold start and sparsity problems in our discussion because both of them primarily deal with lack of information availability. Cold start problems are either caused by new items or new users of the system. For new items there are not enough reviews to recommend them to a user. And for a new user, it is almost impossible to recommend items because there is no history of purchase transactions and the user is new to the system and its difficult to predict his/her taste. Sometimes there is no user profile too. The next biggest challenge is sparsity, which is a subcategory of cold start problem. If a shopping website has lots of items and only a few of them are rated by the users, then there is a review sparsity problem. Sparsity may also introduce bias while recommending items to the users. This problem is also known as lack of knowledge problem because it is hard to recommend an item that is not rated against items that have received lots of ratings.

#### 3.1.2 Over Specialization

In content based filtering recommendation systems, items that are already familiar to the users are recommended. And hence the users are not quite surprised by the recommendations made by the system but users always love surprises. A good recommendation system should always recommend a variety and a wide range of items to the users.

### **3.1.3 Privacy**

In demographic recommendation systems, user's information such as gender, email, age, address, hobbies and sometimes their salary information is required. However, this is dangerous to the users because there is a possibility that the users' information can be hacked and therefore their privacy is breached.

### **3.1.4 Scalability**

As the number of users increase there is a direct demand to increase the number of items available in the shopping website. New resources are added to efficiently determine items with similar characteristics and to make recommendations to users with similar tastes. This is a problem in collaborative filtering technique.

### **3.1.5 Freshness or a Change in Prediction**

There are recommendation systems that can recommend a wide range of items to the users but the users are already familiar with it. For example, recommending new arrivals to the users, recommending top selling items of a shopping website to the user, etc.

## Chapter 4

### 4.1 Handling Cold Start Problems

Most common problem in recommendation systems is the cold start problem. When there is no data available for recommendation the cold start problem occurs. This is broadly classified into cold start user problem and cold start item problem based on which data is missing [6]. We will discuss briefly about them in the next section.

### 4.2 New User Cold Start Problem

Recommendation systems in general are targeted to the individual users and recommend items that they like. So when a user is new, his or her history is not present and user profile data is incomplete or missing. In such scenarios it is impossible to make good recommendations to the new users. Suggesting things that the users may not like may disappoint them. Thus, bad recommendations lead to loss of new users from the system which ideally should not happen in a business point of view. So before even recommending items to the users we should gain enough information about them. There are studies which propose techniques to extract this knowledge from the minimal information provided by the new users. Researchers tried out various approaches in order to solve this problem. Now, we will discuss some of the most common methods proposed in research studies.

#### 4.2.1 Demographic Information

The users have an option of entering data about them which will be a part of their profile

or users can provide their information by signing up using their social media accounts such as Facebook, Twitter or Gmail applications. This will help the recommendation system to know which group the user belongs to. Depending on the type of recommendation system, we can collect new users' information and use this information for recommending items to the users. Similar group of people share similar interests so we can group people by demographic information such as age, gender, zip code, marital status, occupation, student status, education, city, country, salary, number of people in household, number of children in house, number of cars and many more to list. An example as shown in Fig. 9. This information can be gathered by asking users simple questions or from their social media profile. These questions should be simple and should not bore or disappoint new users. Depending on the domain of application questions can be generated.

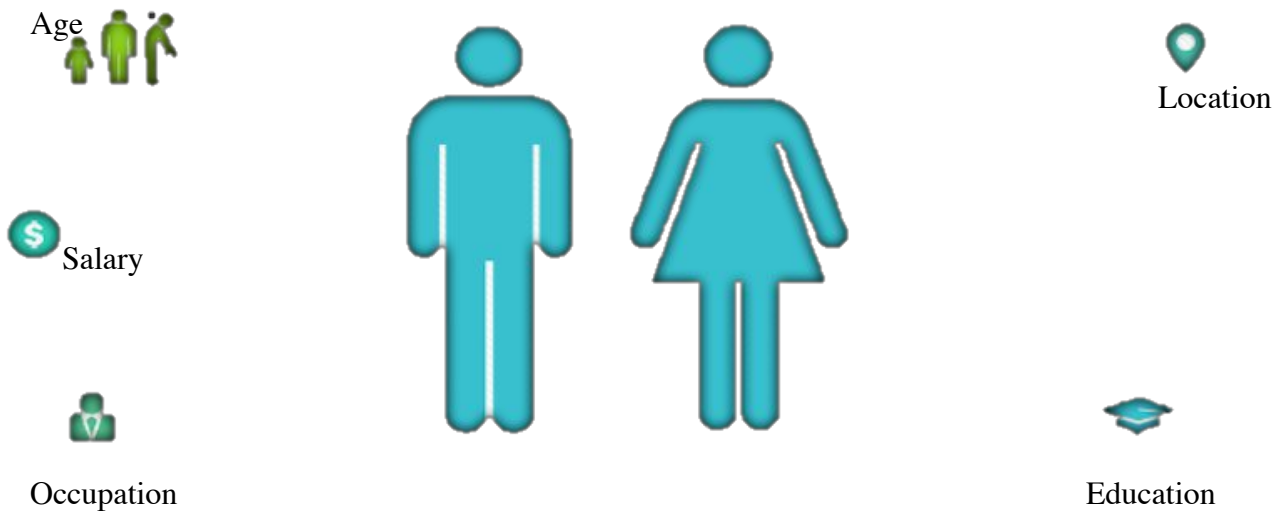


Fig. 9. Demographic information example

If all the information is available through social media profile, the user need not take the trouble to fill the profile information. A demographic recommendation system was implemented on the tourism app called TripAdvisor, a tourist attractions and booking website by Y. Wang et al. [8]. In this paper, they used a crawler to collect demographic data and ratings from TripAdvisor website. The website had information about travelers including travel style, age, gender, travel for, travel with and great vacation attributes. To classify these ratings, Y. Wang et al. used popular machine learning algorithms such as Support Vector Machines (SVM), Naïve Bayes and Bayesian network. SVM model performed the best followed by Naive Bayes model. Bayesian Network underperformed when compared to these two models. All these algorithms were used in recommendations and performed better than their baseline algorithm.

After further inspection of the results, authors of [8] realized that they did not have accurate recommendations for users who rated items between 1-4 and the data was highly imbalanced because majority of the users in TripAdvisor rated their items from 4-5. In future work, the authors said they will come up with a hybrid recommendation technique which will use information about reviews and attractions. There are limitations of using demographic information to solve cold start user problems because they cannot be applied to all the domains in recommendation systems and it is not a general solution for solving cold start problems. Therefore, this technique opened doors for more research in this area.

#### **4.2.2 Requesting Users to Provide Ratings**

In this paper [9], the author says there is a direct way to collect information of new users by asking them questions to rate certain items. However, this is not a great idea because it may be stressful for the new users. So it is very important to ask simple and required questions so that

we can collect only necessary information. These questions will help us in understanding the new users taste, his or her likes and dislikes, and these questions should collect information as much as possible for the recommendation system. The questions asked should not be general and agreeable by everyone since these questions will not distinguish the users. Questions must be formulated in such a way that we should be able to extract information required by the recommendation system. For example, in a food recommendation system, the recommender should not prompt a question like “Do you love chocolate ice cream?” because here almost everyone will answer yes.

So the questions asked to the users must be opinion oriented and questions should not be too specific to the users because this will make the rating process complex. The user may not know how to answer resulting in less knowledge extraction and a bad recommendation system. In paper [9], the authors propose a method to recommend items to a new user using Movie Lens dataset. They figured out four attributes on which the system can be judged on and they are: - a) Users effort – How difficult is the sign up process? b) User satisfaction - Did the user like signing up process? c) Recommendation accuracy – Accuracy of recommendation system in recommending items to the users? d) System utility – Overall, how well did the Recommendation System serve all the users. Effort and accuracy are easy and simple to measure so the researchers chose them as their metric. They came up with five different techniques to present questions to the new users.

#### i) Random

In random technique, items to be presented to the users are picked randomly from a list of items and the users are asked to rate these items. The other way of presenting items to users to

rate them is to pick an item randomly from a list of popular items and then to pick other items randomly. The main advantage of this technique is that it collects a lot of information. The disadvantage of this technique is that since items are presented at random for the users to rate they may not have an opinion on these items.

#### ii) Popularity

In this technique, only the popular items are considered. These items are ranked from most popular to least popular before presenting them to the users to collect ratings. The advantage of this technique is that we can collect the new users' opinion because he or she will definitely give an opinion on popular items. But, since the popular items are very general the recommendation system model may thus become generic and may end up learning nothing new about the new user. Hence, this technique will be of no use at all in solving cold start problems.

#### iii) Pure entropy

In this technique, users are presented with a list of items with very high entropy. This means the items present in the list are highly likable or dislikable by the new users. Items with highest and lowest ratings will have high entropy. So the list of items presented to the users are either rated 5 or 1 on a scale of 1 to 5. The main disadvantage of this technique is that the new users may not have an opinion on the high entropy items at all and will fail to solve our cold start problem.

#### iv) Balanced

Balanced technique is a combination of two techniques discussed previously. Items are selected by combining popularity and pure entropy technique. This technique is most likely to be of benefit because the new users will have an opinion about these items and the recommendation

system can use this information to build a model. This technique is more likely to succeed in collecting users' opinions.

v) Item – Item

In this technique, the items are presented to the new users using one of the above techniques that we discussed before. On the basis of what was rated by the new users, the related items are presented to the users. The next items are chosen as follows: if the same item was previously rated by other users then their other items are presented to the new users for rating. In the next section, we will discuss the results presented in A. M. Rashid et al [9]. Their results are in the Fig. 10.

Strategy	User Effort	Accuracy
Random	★	★★
Popularity	★★★★★	★★★
(log) Pop*Ent	★★★	★★★★★
Item-Item	★★★★★	★★

Fig. 10. Results of A. M. Rashid et al. technique [9]

In their paper [9], the authors have picked user effort and accuracy as their metrics to evaluate their experiment. User effort was measured in terms of how many new users quit during the sign up process. More number of stars mean less user effort and less dropouts. And the other



metric is accuracy, which is a standard metric to measure any machine learning algorithm. For accuracy metric, the authors used Mean Absolute Error (MAE). The text book definition of MAE is the sum of absolute differences between each prediction and the corresponding rating divided by the number of ratings.

What type of rating technique to pick for a recommendation system is dependent on the domain of the application. So before even solving an issue, one has to have collected various information about their domain. Say for example, if we are building a recommendation system for an e-commerce website, then we might probably use popularity technique to select items for the new users to rate because we need to collect information about them as we have no data about them in order to make recommendations. After collecting enough information about the new users we can move on to item – item technique for recommending items to the new users. There may be cases in other domains of recommendation systems where the balanced technique works better because we will be able to collect information about the new users and build a model for their recommendations.

#### **4.2.3 Asking Users to Select Few Users That They Trust**

In [10] the authors propose a method to overcome cold start user problem. The method is as follows: - The new users are given a choice to select existing user or users whom they trust. And this is the beginning point in recommending items to the new users. Later on, the model can be built according to the new users input and feedback. The main issue in this strategy is that the new users have to trust other user or users. Trust is a very crucial factor and we cannot measure the factor of how many new users will be willing to do this and therefore, this technique cannot be a good practical approach.

#### **4.2.4 Tags**

In this section, we will discuss a method proposed by H. Kim et al. [11]. Generally, collaborative filtering technique builds a User X Item matrix but these matrices are typically sparse because it is very difficult for the user to rate all the items present. So the author suggests grouping these items into tags and coming up with a User X Tags matrix for the user. User X Tags matrix are denser than User X Item matrix. This approach proposed by the authors was tested. They used a dataset from del.icio.us, a website containing bookmarks that various users have tagged.

This method was better than the traditional item based collaborative filtering approach and the authors used the recall metric to compare the efficiency of the algorithm with traditional item based collaborative filtering approach. Using this tags model, they generated top – N items for recommendation using Naïve Bayes algorithm. H. Kim et al. said that their method for handling cold start problems has a drawback. If the tags were noisy then the model's performance drastically goes down. In real world scenarios, the tags are created by the users and it is always noisy, so this is a major issue.

#### **4.3 Cold Start Item Problem**

Cold start item problems are very difficult to handle and there are very few approaches proposed in research studies to successfully handle cold start item problem. In this section, we will discuss a few approaches proposed by related research studies to see how they handle cold start item problem.

When there are no ratings on new items, it is impossible to predict ratings of all new items using a traditional collaborative filtering technique [12]. A. I. Schein et al. [12] came up with an answer to this problem. He used probabilistic approach which is a combination of content based filtering and collaborative filtering information by using something known as expectation maximization (EM) [13] making the model learn with the available data and to fit to it. They verified their proposed approach on a movie dataset and they considered the cast attribute of a movie and how similar are they to what the users have rated previously.

The cold start item problem can be easily handled in a content based filtering system when compared to cold start item problem using collaborative filtering approach. In content based filtering system, metadata of items can be used to match against information present in user profile. This works even if the items are new or have never been seen by the users. Ask to rate technique which was described in the previous section will help the users to discover new items if the items presented to the user are randomly selected from list of all items.

## Chapter 5

### 5.1 Deep Learning

Deep learning is a type of machine learning which functions very similar to neurons in a human brain. Deep learning algorithms learn the way our neurons learn and also their structure is a blue print of a neuron too. Combining many of these neuron like structures in a certain fashion results in deep neural networks. Many real time problems that are impossible to solve using machine learning algorithms can be solved using deep learning algorithms. Machine learning and deep learning has been there for a while from now but gained popularity recently due to enormous amount of data which results in bigger models which in turn requires more computation.

Artificial neural network is another name of deep learning algorithm; the name is given due to the similarity in functioning of the neuron [14]. There are many types of networks based on the number of neurons, type of neurons, number of layers, type of connection between the neurons. Perceptron (P), Radial Basis Network (RBN), Feed Forward Network (FFN) as seen in Fig. 11. are few basic artificial neural networks. Convolution Neural Network (CNN), Recurrent Neural Network (RNN) are few complex network examples.

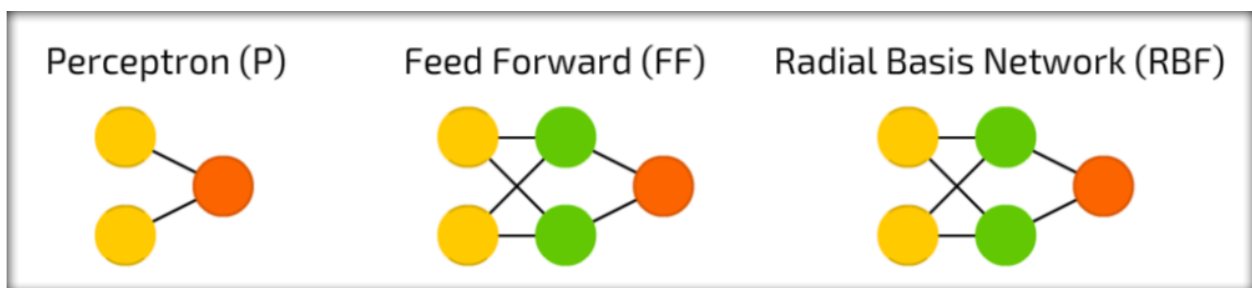


Fig. 11. Example of basic neural networks

## 5.2 Deep Learning Architectures

Deep learning architectures are designed based on the scope of the application. These architectures are arrangement of many basic and complex networks that we discussed in previous section. VGG, AlexNet, Inception-V3 are few popular architectures used widely today. Scientists and researchers carefully design the architecture and this is one of the main contribution for its success. All the implementation details of these architectures are available on World Wide Web. Because of this they are very robust with the help of others' feedback.

### 5.2.1 Applications of Deep Learning Architectures

Deep learning is widely used in all the fields today. To understand more about these applications [15], we can broadly classify them as recognition of pattern and computer vision, computers games, self-driving cars and robotics, acoustical engineering or in simple words creating and managing sound waves, creating art, hallucinations using deep dream and other adventures. Last but not the least, is natural language processing. Coming to future of AI, nobody can predict what may or may not happen. An example of what a computer vision algorithm can do is shown in Fig. 12. It is a multi-object detection algorithm on images.



Fig. 12. An example of multi object detection in computer vision

### 5.3 Visualizing Features of Images

In order to understand how a simple neural network can identify and classify objects of an image, first we need to understand how the neural network sees and works on these images. The authors of paper [16], a group of people in Google brain team tried to visualize features of each layer using their own GoogleNet architecture. After training on ImageNet dataset using GoogleNet, something magical happened. When they tried to visualize what these neurons saw after training was difficult to believe. They saw that neurons are able to remember objects, parts of an object, textures, patterns and edges of an image as shown in Fig. 13. and Fig. 14.

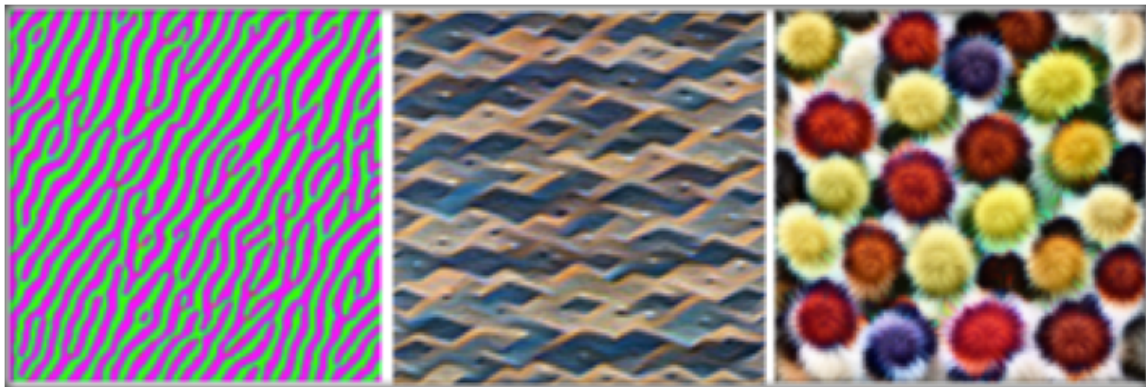


Fig. 13. Example of single neuron feature visualization. From left to right edges, textures and patterns from [16]

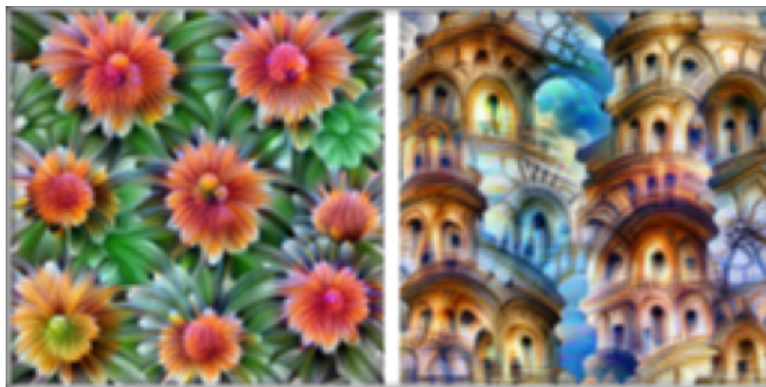


Fig. 14. Example of single neuron feature visualization. From left to right object parts and objects from [16]

The combination of what two neurons saw gave jaw dropping results. When two neurons interacted with each other, they not only remembered things but they also translated/exchanged styles among themselves as shown in Fig. 15. Hence this proves the success of such algorithms in fields like object detection, real time detection and many more. Due to this main reason deep neural networks are being widely combined and used with computer vision applications today.



Fig. 15. Visualization of interaction between two neurons from [16]

Furthermore, to make neural networks more easy for humans to understand, two categories were recently born in the research field. They are feature attribution and visualization. An example is shown in Fig. 16. and Fig. 17. from [16]. Feature attribution research investigates what section of an example image is responsible for activation of the network in a certain way. Feature visualization on an example dataset answers many questions like what are the subsections of a network looking for in an image? And what is the entire network as a whole looking for in an image?



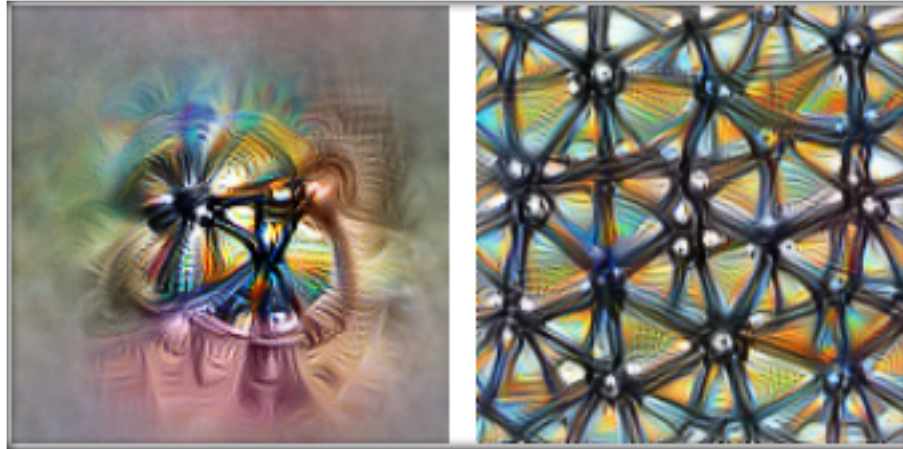


Fig. 16. Example of feature visualization from [16]



Fig. 17. Example of feature attribution from [16]

Using [16] as a proof of concept for this research, we will propose and discuss a technique to solve one of the most common and challenging problems in recommendation systems as of today.



## Chapter 6

### 6.1 Proposed Technique

Consider a new product 'A' in an online shopping website like ebay, whose image and description are available, but reviews not. We can still recommend this product to users by using the image of this product by extracting features using deep learning architecture. We will be able to recommend this product to users who like similar items. This technique will help us to tackle new item cold start problems in recommendation system.

Real time new item cold start problems in general won't have any data to work with, so we will not be using any textual data in this research. Images of new items are always available so will be make use of this to recommend items to the user. Images are downloaded from Google using a custom query API. Proposed recommendation technique aims to build a deep learning model to recommend relevant new items to the users. We will be using VGG 19, a deep learning architecture to extract features from all the images. VGG 19 is made up of CNNs and CNNs are very good at learning and remembering patterns. As images are made up of different patterns and textures, CNNs are best suitable for this task. We will store these extracted features as vectors. Similar items are obtained using K-nearest neighbors algorithm; K is the number of neighbors. Finally, we use a sorting algorithm to rank images from most similar to least similar. We will hyper tune our model using various parameters of K-nearest neighbors algorithm.

In next section, we will be discussing methods for data collection, data processing, deep learning modeling and evaluating the model by comparing it with existing systems.

## 6.2 Data Collection

We will be collecting images for our research by scraping them directly from Web with the help of an API written in python. There are many ways to download images from World Wide Web. We will be using the most common libraries BeautifulSoup and urllibreq for downloading images from a search engine like Google.

For example, if we want to collect images of a whale, we can enter our search variable as whale in the API. The API will go ahead and search for images with whale, construct the URL of the image and download it. Most importantly, we need to make sure that the downloaded images match our search variable to ensure this we will be downloading the top N most relevant images. N can take any numeric value.


We can download N number of relevant images by this method and store it to a file on local disk for further processing.

## 6.3 Data Processing

Data processing is all about making sure we have valid data and relevant data, avoid bias in data, and most importantly ensure that they are consistent with each other. Some of the attributes of an image are its size on local disk, color space, dimensions and type. Size is measured in bytes, color space can be RGB which has three channels or HSV or CMY'K which has a single channel. Dimensions of digital images are its height and width measured in pixels, kind or type of an image is determined by its extensions like GIF or JPEG/JPG or PNG, etc. Alpha channel, another attribute of an image widely used in photo editing in applications like

Photoshop is the fourth channel which is only supported in GIF and PNG and determines how opaque or how transparent an image is. We can use the alpha channel also known as mask channel of a PNG image to remove or add transparency to it. We will briefly discuss these parameters in the following sections. An example image with its attributes is shown in Table 1.

Table 1 Example of image parameters in an image

Image	Image Parameters
	Size: 295 KB
	Type: JPEG
	Alpha Channel: No
	Dimensions: 800 x 1200
	Color Space: RGB

### 6.3.1 Resizing an Image

In order to avoid any sort of bias introduced due to pixel size, we will be resizing all our images in our database to 100 x 100. That is whose height and width is equal to 100 pixels. This can be done by using PIL library, PIL stands for Python Imaging Library. It is an open source library available for python programming to manipulate images and it supports various types of images. It runs on most of the operating systems available today.

During resizing process, an original image will be converted into 100 x 100 pixels. We will also preserve the original color space, type of an image and alpha channel. When the process is completed we will save the images to file on disk for further processing. An example of before and after resizing of an image can be seen in Fig. 18. The other reason for performing resizing of images is to limit the resources required during feature extraction. This will also help us in decreasing computational time.

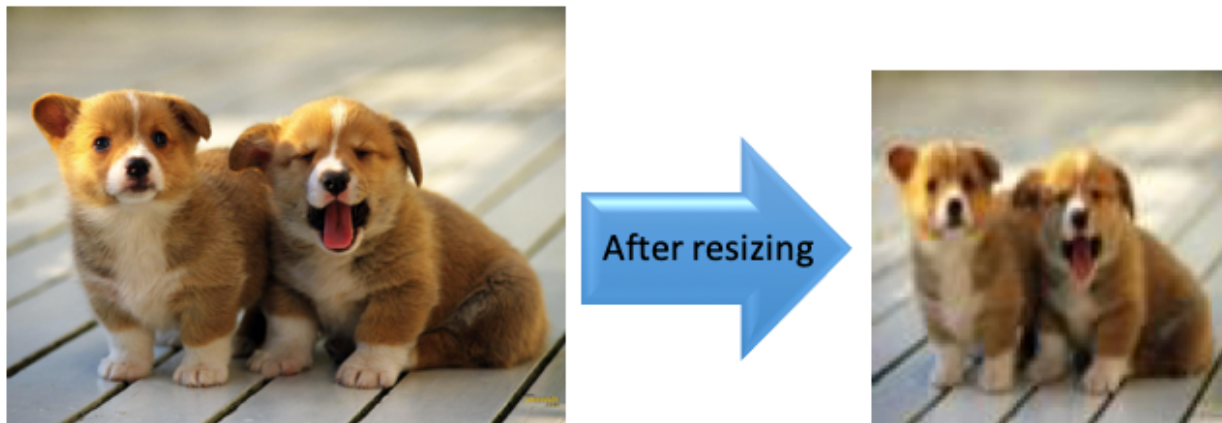


Fig. 18. An example of an image after resizing process

There are various image processing tools to reduce the computational time and power of a deep learning architecture like grey scaling, outline detection, rotate, shear and many more. But in our research we will be performing only resizing on our image dataset.

## 6.4 Model

Choosing a deep learning model to extract features from images for our recommendation

is a crucial task. Many studies in visual recognition field have proved that the state of art is currently been set by convolution neural networks (CNN). For example, in some computer vision tasks like classifying images, humans underperform when compared to the performance of CNN. This means they perform better in visual tasks than humans. In 2014, a team of researchers from Oxford University designed a model and participated in ImageNet large scale visual recognition challenge. Their model performed extraordinarily well and was designed purely using CNN. They secured first prize for image localization task of finding a particular object in an image and drawing a bounding box around it. They secured second place for image classification task - a task of identifying object of an image and describing them with a label [17]. This challenge used a dataset called ImageNet. It is a huge visual dataset which consists more than 1000 classes and is widely used in academic researches [18].

The Oxford University team that won the challenges and their models widely became known to everyone. They were from the popular group called visual geometry team. So they decided to name their models that won the challenges as VGG models. They designed two versions of it depending on the number of layers present in it. The 16 layer model has 16 layers in it and the 19 layer model has 19 layers in it [17]. The 19 layer model outperformed the 16 layer model in the competition. This is the reason for picking the 19 layer model to perform our experiments. The VGG made their models public in 2015 to help it get better and to help researchers in their research. They also designed a third version of VGG model named hybrid VGG model in 2015. The hybrid model known as fusion performed better than VGG16 and VGG19 with least training and testing errors but they did not say anything more about this model.

### **6.4.1 Architecture of VGG19**

The architecture design of VGG16 and VGG19 was inspired by the architecture of AlexNet. The designers of VGG models replaced the large convolution filters in AlexNet with many small 3 x 3 convolution filters one after the other. Authors of [19] also confirm that these 3 x 3 convolution filters helped in preserving finer details of the image. And worked in combination with multiple max pooling layer, softmax layers and fully connected layers. In 2014, these architectures were considered to be very deep. In order to train these large architectures, the VGG team first trained smaller networks with lesser weighted layers [19]. So this was known as pre-training process and it was used as initiators for VGG16 and VGG19 models. The pre-training itself took a lot of time and was a tedious task. Then the authors of [19] used this small pre-trained network to train the entire network.

Along with its advantages, VGG comes with many disadvantages. It is very difficult and slow to train the network. The architecture of this network has very large weights. Due to the presence of fully connected layers and deep depth, the trained network weight itself is about 540MB and 580MB for VGG 16 and VGG 19 models respectively. In spite of all these disadvantages VGG is a widely used model even today but smaller architectures like GoogleNet is more desirable. VGG 16 alone has a total of about 138.5 Million parameters.

### **6.4.2 Transfer Learning**

Transfer learning is a popular technique in machine learning where a certain model is developed to solve a particular problem and can be reused as an initiator to solve a different task. They are mostly used in natural language processing and computer vision applications [20]. Transfer learning is also known as optimization technique because it is already trained to solve a

task and often speeds up the progress of its related task. There are two approaches in transfer learning, develop model technique and pre-trained model technique. The develop model transfer learning technique is not widely used in deep learning so we will discuss about it.

Pre-trained model transfer technique is widely used because not all research studies can afford 2-3 years of time and the high cost just to train a deep learning model. In this technique, we pick a model trained to solve a common task like classifying images. Then we reuse the entire model or use a part of it to solve the new task [20]. Depending on the task we can optionally tune the model.

In this research we will be using a part of VGG 19 network as an initiator for our recommendation task. As the higher level layers are used for classification task in this model, we will use only the low level layers to read all the images in our database and convert images to a vector containing features. Then using K nearest neighbors' algorithm, we find K nearest neighbors for every image in the database and store the recommended images to a file for future use.

## Chapter 7

### 7.1 Baseline for Evaluation

In [21], the authors developed a model to address cold start problems in social networking websites like etsy and pinterest using images liked by the users. They also collected contextual information of new items. Which constitutes of visual, topic and spatial data. Their model used implicit feedback provided by the users to improve the performance. They trained their model on Flickr dataset. During their course of experiments the highest recall was 0.0769 and precision was 0.0893 at  $K = 5$ . So we will be considering results of [21] as baseline for our experiment.

Table 2 Baseline for our research

Author names	Results @ k=5
Niu, W. et al. [21]	Recall: 0.0769 Precision: 0.0893

### 7.2 Evaluation Metric

The best possible way to evaluate supervised and unsupervised learning algorithm is by calculating their precision, recall and F score. The values are between 0 to 1. If a value is close to 1 the better it is. Formulae for calculating precision, recall and F score is shown in the Figure 19.



$$Precision = \frac{t_p}{t_p + f_p}$$

$$Recall = \frac{t_p}{t_p + f_n}$$

$$F_1 = \left( \frac{recall^{-1} + precision^{-1}}{2} \right)^{-1} = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

Fig. 19. Evaluation Formulae

In the next section we will discuss more about the datasets, experiment and results.

### 7.3 Hardware and Software Requirements

There are no hard hardware requirements in our research. In order to implement this research we suggest these minimum hardware requirements.

- ❖ Memory: 8 GB
- ❖ Processor: 2.3 GHz Intel Core i7

Here is the list of software resources required to implement this research.

- ❖ Python 3
- ❖ Anaconda-Navigator an IDE to analyze data
- ❖ And create your own database with custom Google API

- ❖ Any OS can be used that can support Python 3
- ❖ Libraries such as Scikit Learn or SK Learn, Tensor Flow and Keras

All of the above listed softwares are required and is free to download.

## 7.4 Dataset 1

Our dataset 1 consists of four different shapes in geometry (circle, cylinder, triangle and square). We have 10 images of each shape. All the images were downloaded from Google using a custom API that we discussed in the previous chapter. We downloaded only JPEG images because these images are universally accepted. All the images were resized to 100 x 100 pixels in data preprocessing phase. We preserved their original color space. Further, the feature vectors were extracted from all the images using VGG 19 architecture and flattened for comparison using KNN algorithm. Finally, we used a sorting algorithm to rank the images from most similar to least similar images. Glimpse of our dataset 1 can be seen in Figure 20.

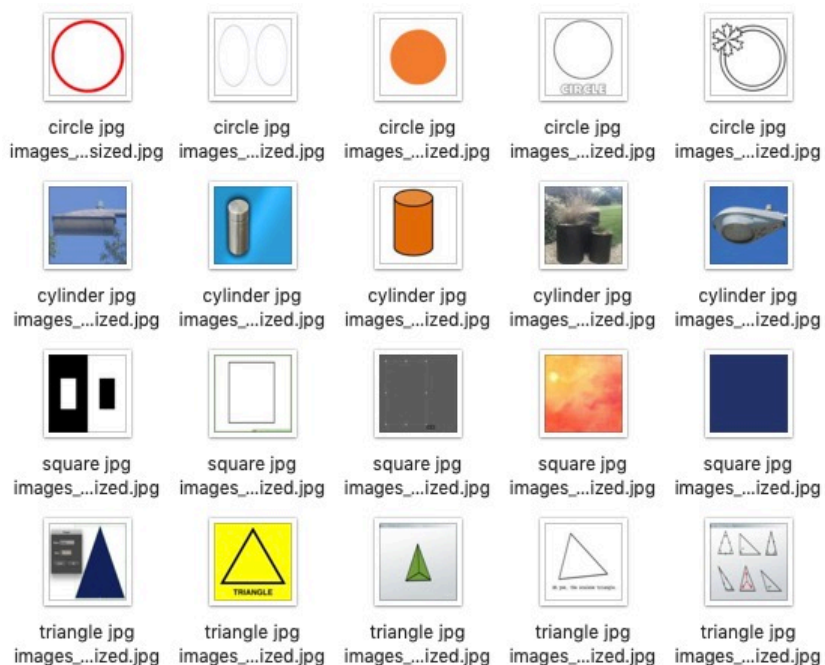


Fig. 20. Glimpse of Dataset 1

We experimented with different values of K in KNN algorithm and hyper tuned the model using hyper tuning parameters such as algorithm, metric in KNN algorithm. Weights parameter is set to uniform which is a default value. We will briefly discuss about hyper tuning in further sections. The results for dataset 1 are tabulated in Table 3. The results were averaged over 40 runs for each combination of K and hyper parameter values.

Table 3 Experimental results of Dataset 1

<b>K values</b>	<b>Algorithm = brute Metric = cosine Weights = uniform</b>	<b>Algorithm = auto Metric = cosine Weights = uniform</b>	<b>Baseline results @ K = 5</b>
<b>K = 1</b>	Precision: 0.5125 Recall: 0.2875 F Score: 0.3682	Precision: 0.5100 Recall: 0.2850 F Score: 0.3655	Precision: 0.0893 [21] Recall: 0.0769 [21]
<b>K = 2</b>	Precision: 0.5250 Recall: 0.2900 F Score: 0.3734	Precision: 0.5200 Recall: 0.2900 F Score: 0.3703	
<b>K = 3</b>	Precision: 0.5430 Recall: 0.2995 F Score: 0.3859	Precision: 0.5530 Recall: 0.2995 F Score: 0.3885	
<b>K = 4</b>	Precision: 0.5800 Recall: 0.2900 F Score: 0.3866	Precision: 0.5800 Recall: 0.2900 F Score: 0.3866	
<b>K = 5</b>	Precision: 0.5950 Recall: 0.2975 F Score: 0.3966	Precision: 0.5900 Recall: 0.2950 F Score: 0.3933	
<b>K = 6</b>	Precision: 0.5500 Recall: 0.3300 F Score: 0.4125	Precision: 0.5500 Recall: 0.3300 F Score: 0.4125	

As we can see from Table 3, our precision and recall values are better than the baseline results which is a positive sign. We also calculated F score value as precision and recall values were available. We observed that as the K value increased we saw an increase in precision and recall values. Precision and recall values almost became constant at K=4, K=5 and K=6 in our experiment. We performed hyper tuning on KNN algorithm to improve our results. Parameters considered for hyper tuning were algorithm, n\_neighbors aka K and distance metric.

At K = 5, our baseline precision, recall values were 0.0893, 0.0769 respectively and proposed approach precision, recall values are 0.5950, 0.2975. Our model performed better than baseline model due to following reasons: We have a consistent dataset which consists of simple geometry shapes. The most important reason is we used VGG 19 architecture in our feature extraction phase. Which is trained on ImageNet dataset. ImageNet is a very large dataset. It consists of variety of images. VGG 19 architecture is well known for learning and remembering visual features. Hence the results. The glimpse of our visual recommendations of dataset 1 is shown in Figure 21.

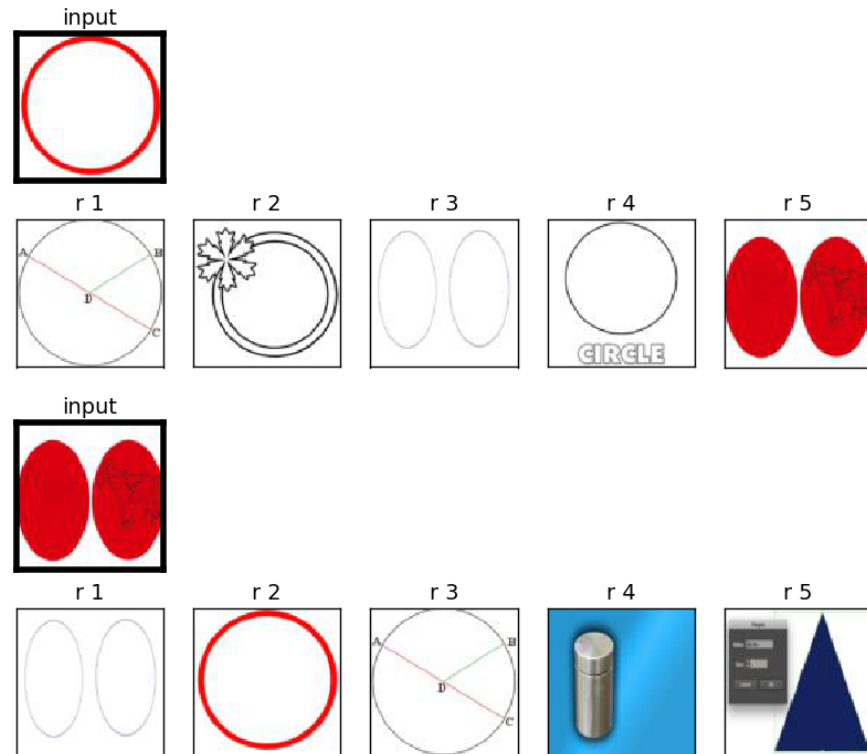


Fig. 21. From top to bottom, the best and worst circle image recommendations given by our model with  $k=5$ , algorithm=brute and metric=cosine

## 7.5 Dataset 2

Our dataset 2 consists of four different junk food items (burger, chicken wings, fries and onion rings). We have 10 images of each food item. All the images were downloaded from Google using a custom API that we discussed in the previous chapter. We downloaded only JPEG images because they are universally accepted. Images were resized to 100 x 100 pixels in data preprocessing phase. We preserved their original color space. Further, the feature vectors were extracted from all the images using VGG 19 architecture and flattened for comparison using KNN algorithm. Finally, we used a sorting algorithm to rank the images from most similar to least similar images. Glimpse of our dataset 2 can be seen in Figure 22.



Fig. 22. Glimpse of Dataset 2

We performed experiments with different values of K in KNN algorithm and also hyper tuned the model using parameters such as algorithm and metric in KNN algorithm. Weights parameter is set to uniform which is a default value. We will briefly discuss about hyper tuning in further sections. The results for dataset 2 are tabulated in Table 4. The results were averaged over 40 runs for each combination of K and hyper parameter values.

Table 4 Experimental results of Dataset 2

<b>K values</b>	<b>Algorithm = brute Metric = cosine Weights = uniform</b>	<b>Algorithm = auto Metric = cosine Weights = uniform</b>	<b>Baseline results @ K = 5</b>
<b>K = 1</b>	Precision: 0.3050 Recall: 0.1625 F Score: 0.2096	Precision: 0.3025 Recall: 0.1875 F Score: 0.2285	Precision: 0.0893 [21] Recall: 0.0769 [21]
<b>K = 2</b>	Precision: 0.3350 Recall: 0.1825 F Score: 0.2361	Precision: 0.3325 Recall: 0.1875 F Score: 0.2384	
<b>K = 3</b>	Precision: 0.3550 Recall: 0.1850 F Score: 0.2429	Precision: 0.3525 Recall: 0.2075 F Score: 0.2610	
<b>K = 4</b>	Precision: 0.3650 Recall: 0.1825 F Score: 0.2432	Precision: 0.3625 Recall: 0.2175 F Score: 0.2717	
<b>K = 5</b>	Precision: 0.3650 Recall: 0.1825 F Score: 0.2432	Precision: 0.3625 Recall: 0.2175 F Score: 0.2717	
<b>K = 6</b>	Precision: 0.3650 Recall: 0.1825 F Score: 0.2432	Precision: 0.3625 Recall: 0.2175 F Score: 0.2717	

As we can see from Table 4, our precision and recall values are better than the baseline results which is a positive sign. We also calculated F score value as precision and recall values were available. We observed that as the K value increased we saw an increase in precision and recall values. Precision and recall values became constant for K = 4, K = 5 and K = 6. We

performed hyper tuning on KNN algorithm to improve our results. Parameters considered for hyper tuning were algorithm, n\_neighbors aka K and distance metric.

At  $K = 5$ , our baseline precision, recall values were 0.0893, 0.0769 respectively and proposed approach precision, recall values are 0.3650, 0.1825. Our model performed better than baseline model due to following reasons: We have a consistent dataset which consists of junk food items. The most important reason is we used VGG 19 architecture in our feature extraction phase. Which is trained on ImageNet dataset. ImageNet is a very large dataset consists of variety of images. VGG 19 architecture is well known for learning and remembering visual features. Hence the results. The glimpse of our visual recommendations of dataset 2 is shown in Figure 23.

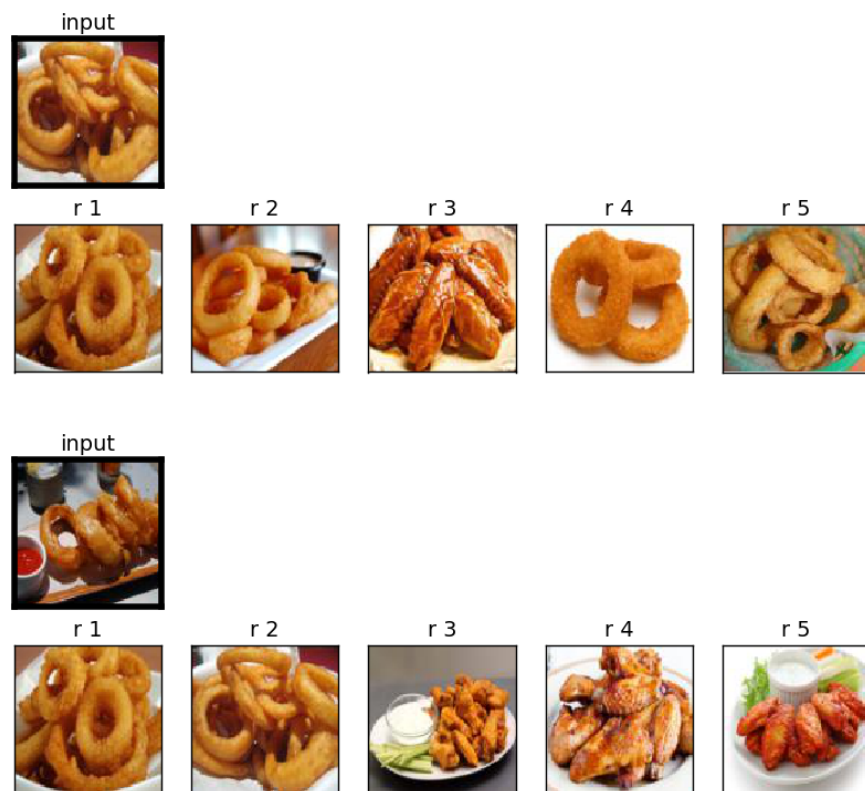


Fig. 23. From top to bottom, the best and worst onion rings image recommendations given by our model with  $k=5$ , algorithm=brute and metric=cosine



## 7.6 Dataset 3

Our dataset 3 consists of two popular junk food items (burger and hotdog). We have 20 images of each food item. All the images were downloaded from Google using a custom API that we discussed in previous chapter. We downloaded only JPEG images because they are universally accepted. Images were resized to 100 x 100 pixels in data preprocessing phase. We preserved their original color space. Further, the feature vectors were extracted from all the images using VGG 19 architecture and flattened for comparison using KNN algorithm. Finally, we used a sorting algorithm to rank the images from most similar to least similar images.

Glimpse of our dataset 3 can be seen in Figure 24.



Fig. 24. Glimpse of Dataset 3

We performed experiments with different values of K in KNN algorithm and also hyper tuned the model using parameters such as algorithm and metric in KNN algorithm. Weights parameter is set to uniform which is a default value. We will briefly discuss about hyper tuning in further sections. The results for dataset 3 are tabulated in Table 5. The results were averaged over 40 runs for each combination of K and hyper parameter values.

Table 5 Experimental results of Dataset 3

<b>K values</b>	<b>Algorithm = brute Metric = cosine</b>	<b>Algorithm = auto Metric = cosine</b>	<b>Baseline results @ K = 5</b>
<b>K = 1</b>	Precision: 0.6250 Recall: 0.3400 F Score: 0.4404	Precision: 0.6300 Recall: 0.3450 F Score: 0.4457	Precision: 0.0893 [21] Recall: 0.0769 [21]
<b>K = 2</b>	Precision: 0.7200 Recall: 0.3600 F Score: 0.4800	Precision: 0.7200 Recall: 0.3600 F Score: 0.4800	
<b>K = 3</b>	Precision: 0.7450 Recall: 0.3725 F Score: 0.4966	Precision: 0.7450 Recall: 0.3725 F Score: 0.4966	
<b>K = 4</b>	Precision: 0.7450 Recall: 0.3725 F Score: 0.4966	Precision: 0.7500 Recall: 0.4500 F Score: 0.5625	
<b>K = 5</b>	Precision: 0.7500 Recall: 0.4500 F Score: 0.5625	Precision: 0.7600 Recall: 0.3800 F Score: 0.5066	
<b>K = 6</b>	Precision: 0.7450 Recall: 0.3725 F Score: 0.4966	Precision: 0.7600 Recall: 0.3800 F Score: 0.5066	

As we can see from Table 5, our precision and recall values are better than baseline results which is a positive sign. We also calculated F score value as precision and recall values were available. We observed that as the K value increased we saw an increase in precision and recall values. Precision and recall values became constant for K values 3, 4, 5 and 6. We performed hyper tuning on KNN algorithm to improve our results. Parameters considered for hyper tuning were algorithm, n\_neighbors aka K and distance metric.

At  $K = 5$ , our baseline precision, recall values were 0.0893, 0.0769 respectively and proposed approach precision, recall values are 0.7600, 0.3800. Our model performed better than the baseline model due to following reasons: We have a consistent dataset which consists of junk food items. The most important reason is we used VGG 19 architecture in our feature extraction phase. Which is trained on ImageNet dataset. ImageNet is a very large dataset consists of variety of images. VGG 19 architecture is well known for learning and remembering visual features. Hence the results. The glimpse of our visual recommendations of dataset 3 is shown in Figure 25.

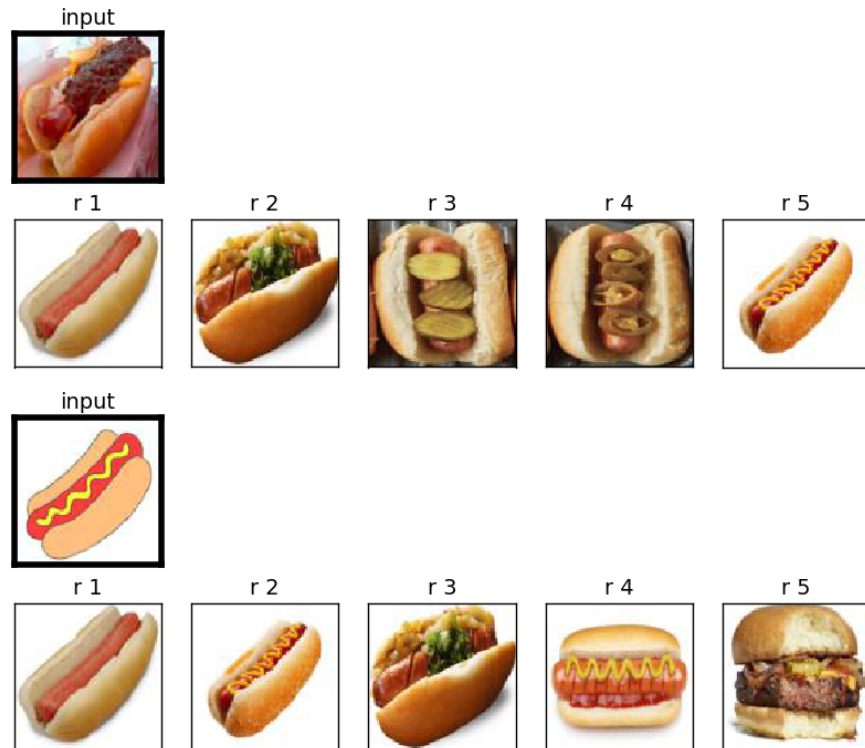


Fig. 25. From top to bottom, the best and worst hotdog image recommendations given by our model with  $k=5$ , algorithm=auto and metric=cosine

## 7.7 Dataset 4

Our dataset 4 consists of four different types of landscapes (desert, glacier, coast and taiga). We have 10 images of each landscape. All the images were downloaded from Google using a custom API that was discussed in previous chapter. We downloaded only JPEG images because they are universally accepted. Images were resized to 100 x 100 pixels in data preprocessing phase. We preserved their original color space. Further, the feature vectors were extracted from all the images using VGG 19 architecture and flattened for comparison using KNN algorithm. Finally, we used a sorting algorithm to rank the images from most similar to least similar images. Glimpse of our dataset 4 can be seen in Figure 26.

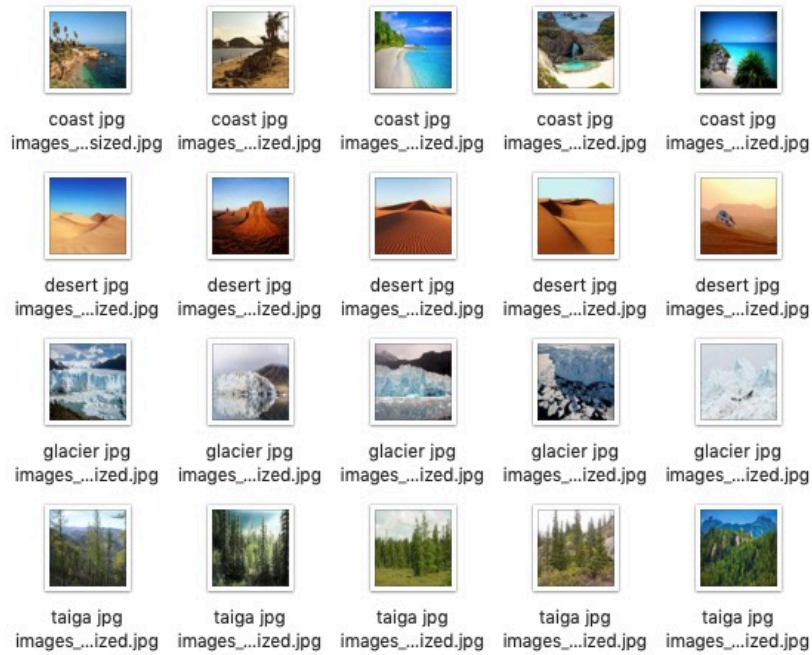


Fig. 26. Glimpse of Dataset 4

We performed experiments with different values of K in KNN algorithm and also hyper tuned the model using parameters such as algorithm and metric in KNN algorithm. Weights parameter is set to uniform which is a default value. We will briefly discuss about hyper tuning in further sections. The results for dataset 4 are tabulated in Table 6. The results were averaged over 40 runs for each combination of K and hyper parameter values.

Table 6 Experimental results of Dataset 4

<b>K values</b>	<b>Algorithm = brute Metric = cosine</b>	<b>Algorithm = auto Metric = cosine</b>	<b>Baseline results @ K = 5</b>
<b>K = 1</b>	Precision: 0.4550 Recall: 0.2550 F Score: 0.3267	Precision: 0.5000 Recall: 0.2555 F Score: 0.3381	Precision: 0.0893 [21] Recall: 0.0769 [21]
<b>K = 2</b>	Precision: 0.4550 Recall: 0.2550 F Score: 0.3267	Precision: 0.5000 Recall: 0.2555 F Score: 0.3381	
<b>K = 3</b>	Precision: 0.5500 Recall: 0.2725 F Score: 0.3642	Precision: 0.5500 Recall: 0.2725 F Score: 0.3642	
<b>K = 4</b>	Precision: 0.5650 Recall: 0.2825 F Score: 0.3766	Precision: 0.5375 Recall: 0.3225 F Score: 0.4030	
<b>K = 5</b>	Precision: 0.5650 Recall: 0.2825 F Score: 0.3766	Precision: 0.5750 Recall: 0.2900 F Score: 0.3854	
<b>K = 6</b>	Precision: 0.5375 Recall: 0.3225 F Score: 0.4030	Precision: 0.5375 Recall: 0.3225 F Score: 0.4030	

As we can see from Table 6, our precision and recall values are better than the baseline results which is a positive sign. We also calculated F score value as precision and recall values were available. We observed that as the K value increased we saw an increase in precision and recall values. Precision and recall values almost become constant for K values 4, 5 and 6. We

performed hyper tuning on KNN algorithm to improve our results. Parameters considered for hyper tuning were algorithm, n\_neighbors aka K and distance metric.

At  $K = 5$ , our baseline precision, recall values were 0.0893, 0.0769 respectively and proposed approach precision, recall values are 0.5750, 0.2900. Our model performed better than baseline model due to following reasons: We have a consistent dataset which consists of different landscapes. The most important reason is we used VGG 19 architecture in our feature extraction phase. Which is trained on ImageNet dataset. ImageNet is a very large dataset consists of variety of images. VGG 19 architecture is well known for learning and remembering visual features. Hence the results. The glimpse of our visual recommendations of dataset 4 is shown in Figure 27.

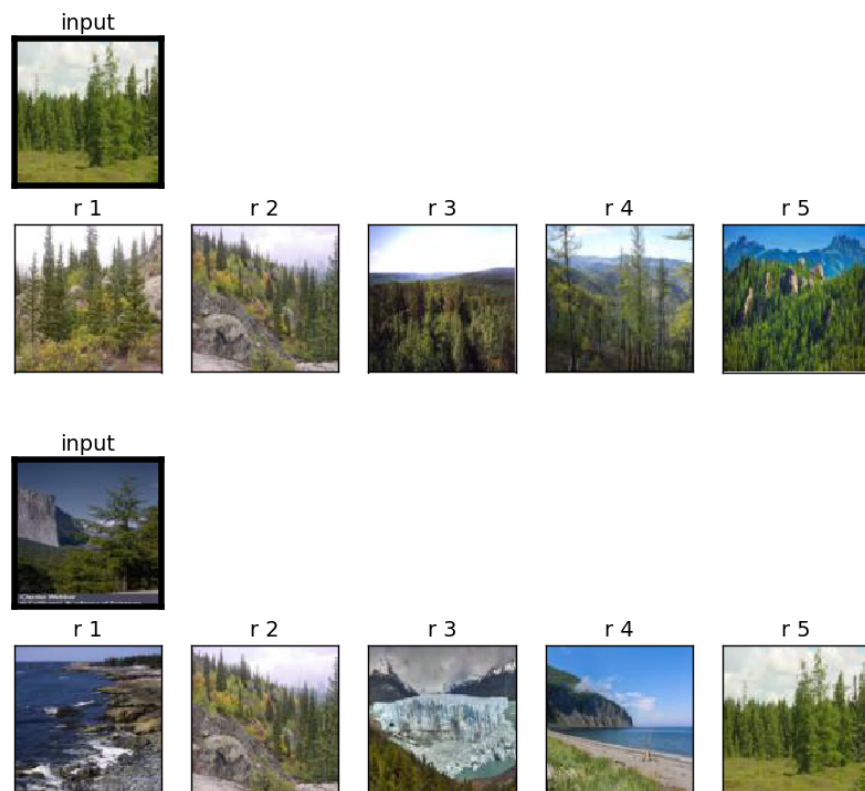


Fig. 27. From top to bottom, the best and worst taiga image recommendations given by our model with  $k=5$ , algorithm=auto and metric=cosine

## Conclusion

In this research, we reviewed different types of recommendation systems, discussed various challenges faced by them and we made an attempt to address the most important challenge. We also discussed several research studies that proposed techniques to handle new item cold start problems. Finally, we proposed a solution as one of the ways to tackle new item cold start problem.

Table 7 Comparison of our experiments with the baseline approach

K value	Baseline	Dataset 1	Dataset 2	Dataset 3	Dataset 4
<b>K = 5</b>	Precision: 0.0893 [21] Recall: 0.0769 [21]	Precision: 0.5950 Recall: 0.2975 F Score: 0.3966	Precision: 0.3650 Recall: 0.1825 F Score: 0.2432	Precision: 0.7600 Recall: 0.3800 F Score: 0.5066	Precision: 0.5750 Recall: 0.2900 F Score: 0.3854

We performed experiments on four different image datasets containing variety of shapes patterns, colors and textures. Precision, recall and F Score are tabulated as seen in Table 7. It contains best results of each dataset @ K = 5 as the baseline results were recorded at K @ 5. We saw better results with a geometric shapes dataset when compared to a junk food dataset. This may be because it had less number of colors and shapes to deal with. In junk food dataset we noticed some images had additional items like fries and coke along with the item of interest like burger and chicken wings. This will confuse the model and it will recommend a fries image for an input containing fries and burger in the same image and vice versa. Hence we saw lowest precision and recall values among our experiments. The hotdog-burger dataset gave us the best results. This may be because the images had only hotdogs or burgers no additional items were



present in the image that would confuse the model to distinguish between burger and hotdog during recommendation. Additionally we had 20 images of each food item whereas in other dataset we had 10 images of each item. More is better.

Finally, landscape dataset performed better than expected because it had to deal with common regions like sand, rock formations and shrubs in the landscape images. The most challenging thing for the model in this dataset may have been to distinguish between the water body present in glacier and coastal landscape image as it constituted larger percentage in coastal and glacier images. Same challenge for sandy regions present in desert and coastal images.

The efficiency of a recommendation system is purely dependent on the data available to it. Good data yields better recommendations and hence better precision, recall and F score values. In the future, we can possibly improve the quality of our recommendations by following these techniques: generating captions for images this could help us recommend relevant items to the users, for example - desert images will have captions such as dry, sand and sun. Whereas coastal images will have captions such as water, sand, cliff. These captions will help in improving our results or we can assign tags on images of items. This could help us to classify item images implicitly, which in turn, will help in improving our results.

## References

- [1] S. Stumpf and S. Muscroft, “When users generate music playlists: When words leave off, music begins?,” presented at IEEE Int. Conf. of Multimedia and Expo (ICME), Barcelona, Spain, 2011.
- [2] J. Lee et al., “Music recommendation system based on usage history and automatic genre classification,” presented at IEEE Int. Conf. on Consumer Electronics (ICCE), Las Vegas, NV, USA, 2015.
- [3] S. Thangavel, P. Bkaratki and A. Sankar, “Student placement analyzer: A recommendation system using machine learning,” presented at the 4th Int. Conf. on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, 2017.
- [4] S. Park, S. Lee and S. Lee, “Session-based collaborative filtering for predicting the next song,” presented at First ACIS/JNU Int. Conf. on Computers, Networks, Systems and Industrial Engineering, Jeju Island, South Korea, 2011.
- [5] S. Gupta, S. Goel, “Handling user cold start problems using fuzzy clustering”, in International conference on ICT for sustainable development (ICT4SD 2016), Panaji, Goa, 2016.
- [6] S.-T. Park, D. Pennock, O. Madani, N. Good, and D. DeCoste. Naïve filterbots for robust cold-start recommendations. In Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 699–705. ACM, 2006.
- [7] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, “Item-based collaborative filtering recommendation algorithms”, in Proceedings of the 10th International Conference on World Wide Web, WWW ‘01, New York, NY, USA: ACM, pp. 285–295, 2001.

- [8] Y. Wang, S. C.-f. Chan, and G. Ngai. Applicability of demographic recommender system to tourist attractions: A case study on trip advisor. In Proceedings of the The 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology-Volume 03, pages 97–101. IEEE Computer Society, 2012.
- [9] A. M. Rashid, I. Albert, D. Cosley, S. K. Lam, S. M. McNee, J. A. Konstan, and J. Riedl. Getting to know you: learning new user preferences in recommender systems. In Proceedings of the 7th international conference on Intelligent user interfaces, pages 127 -134. ACM, 2002.
- [10] P. Massa and B. Bhattacharjee. Using trust in recommender systems: an experimental analysis. In Trust Management, pages 221–235. Springer, 2004.
- [11] H.-N. Kim, A.-T. Ji, I. Ha, and G.-S. Jo. Collaborative filtering based on collaborative tagging for enhancing the quality of recommendation. *Electronic Commerce Research and Applications*, 9(1):73–83, 2010.
- [12] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock. Methods and metrics for cold-start recommendations. In Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval, pages 253–260. ACM, 2002.
- [13] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.
- [14] Chart of neural networks [Online]. Available: [https:// towardsdatascience.com/ the -mostly -complete -chart -of -neural -networks -explained -3fb6f2367464](https://towardsdatascience.com/the-mostly-complete-chart-of-neural-networks-explained-3fb6f2367464)
- [15] Applications of deep learning [Online]. Available: [http:// www.yaronhadad.com/ deep -learning -most -amazing -applications/ #future](http://www.yaronhadad.com/deep-learning-most-amazing-applications/#future)

- [16] Feature visualization [Online]. Available: [https:// distill.pub/ 2017/ feature -visualization/](https://distill.pub/2017/feature-visualization/)
- [17] VGG Model [Online]. Available: [http:// www.robots.ox.ac.uk/ ~vgg/ research/ very \\_deep/](http://www.robots.ox.ac.uk/~vgg/research/very_deep/)
- [18] ImageNet Wikipedia [Online]. Available: [https:// en.wikipedia.org/ wiki/ ImageNet](https://en.wikipedia.org/wiki/ImageNet)
- [19] Very deep convolutional networks for large-scale image recognition [Online]. Available: [https:// arxiv.org/ pdf/ 1409.1556.pdf](https://arxiv.org/pdf/1409.1556.pdf)
- [20] Transfer learning [Online]. Available: [https:// machinelearningmastery.com/ transfer - learning -for -deep -learning/](https://machinelearningmastery.com/transfer-learning-for-deep-learning/)
- [21] Neural Personalized Ranking for Image Recommendation [Online]. Available: [http:// faculty.cse.tamu.edu/ caverlee/ pubs/ niu18wsdm.pdf](http://faculty.cse.tamu.edu/caverlee/pubs/niu18wsdm.pdf)